

REMARKS

Claims 1-20 are canceled. Claims 21-24 were previously canceled.

Claims 25 – 48 are added.

Claims 25 - 48 are presently under consideration.

A document entitled **VERSION SHOWING CHANGES** is attached, showing the amendments.

The amendment to the specification

The specification is amended to be more consistent with current terminology, and to correct minor informalities. No new matter has been added.

New claims 25 - 48

New claims 25 – 48 are added to provide more complete coverage for the present invention. Support for the language of the claims may be found, for example, in the original claims 1-24, as well as at page 8, lines 3 - 8 and page 11, lines 11-13 of the present application.

CONCLUSION

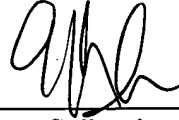
Should the Examiner feel that there are any issues outstanding after consideration of this response, the Examiner is invited to contact Applicants' undersigned representative to expedite the prosecution.

EXCEPT for issue fees payable under 37 C.F.R. § 1.18, the Commissioner is hereby authorized by this paper to charge any additional fees during the entire pendency of this application including fees due under 37 C.F.R. §§ 1.16 and 1.17 which may be required, including any required extension of time fees, or credit any overpayment to Deposit Account 50-0653. This paragraph is intended to be a **CONSTRUCTIVE PETITION FOR EXTENSION OF**

TIME in accordance with 37 C.F.R. § 1.136(a)(3).

Respectfully submitted,

GREENBERG TRAURIG, LLP



By: _____

George S. Bardmesser

Reg. No. 44,020

Dated: August 14, 2001

Greenberg Traurig, LLP

1750 Tysons Blvd.,

Suite 1200

Tysons Corner, VA 22102

703-749-1389

VERSION SHOWING CHANGES**IN THE SPECIFICATION:**

Please amend the paragraphs at page 11, line 10, through page 11, line 26 to read as follows:

--Further, open-ended inviting (**experimentation**) is experimental based and recursive. Using open-ended inviting, several identities (**concept representatives**) are invited to be used with the root node. For each identity invited, an experiment is performed to ascertain whether the identity is appropriate for use with the root node, in part according to the identity's children nodes. Each of the experimenting identities invites other identities for use with the children nodes below it. The children nodes created below the experimenting root node also use open-ended inviting for selecting identities to be used therewith. This recursion continues until it can be determined whether the initial experimenting identity can be used with the root node. Once one is found, children of the root node are established, and open-ended inviting continues on for the children nodes, and the grandchildren nodes, and so on. **Depending on the embodiment of the invention, the recursion can be performed once or a multiple number of times.**

In a preferred embodiment, this recursion continues separately down each branch of the developing SMT, and stops whenever a leaf is reached or a node fails to be satisfied. **(An SMT may be referred to as a "source tree.")** As the recursion unwinds up each branch, **results are** [success or failure is] transmitted up the SMT. [If a node receives "failure" from any of its children nodes, the experimenting node is confirmed as failing.] **If a node receives "success" from each of its children nodes, the experimenting identity at that node is confirmed as successful, unless further tests required by the experimenting identity fail. Regardless of the results from any of its child nodes,**

the experimenting node then proceeds to combine them into a single result for the experiment

above.—

Please amend the paragraph at page 13, line 5, through page 13, line 9 to read as follows:

--Besides discontinuous portions of the source, continuous portions of the source can also be used as the source identifier 116. For example, identities for the source identifiers 116 of "What time is it[?]" and "What is the time[?]" can exist. These identities are invited to be part of the identities list when the source is, respectively, "What time is it now?" and "What is the time in Berlin?".--

Please amend the paragraph at page 13, line 17, through page 13, line 24 to read as follows:

--Another example of an invisible identity available under certain circumstances is the one available when "for whom" is included in a source of text. Not only is there an identity with the source identifier 116 of "for whom," but there is also an invisible identity which is available when "for whom" occurs in a source of text. This invisible identity corresponds to text such as "all men older than 50" and treats the text as though it were "all men for whom their age is greater than 50."

--

Please amend the paragraph at page 16, line 17, through page 17, line 21 to read as follows:

--In block 31, an error message is passed to the user that the source was unable to be

translated and comprehended. Alternatively, with the error message, the user can be prompted to modify the source or input a new source. This alternative is indicated by the dashed flow line from block 31 in Fig. 8. Alternatively, when an experiment fails, or produces an ambiguity, the experimenting identity could determine why the experiment failed, or decide between the alternative results as a further problem to be solved, and apply open-ended inviting again to solve this further problem. To solve this further problem, the experimenting identity could enlist the knowledge in the system to determine what the user intended to say, or really meant.--

Please amend the paragraph at page 17, line 18, through page 17, line 21 to read as follows:

--In block 13, it is determined whether any source is remaining to be matched. If all the source has been matched, the flow proceeds to block 9 in Fig. [6]5. When this occurs, the entire source has been replaced by a SMT. If there is still source remaining to be matched, the flow proceeds to block 14.--

Please amend the paragraph at page 22, line 24, through page 23, line 2 to read as follows:

--In block 54, the implementation of lists of identities is determined. The identities in the subtree that self-manipulates in block 54 replace themselves and their child subtrees with subtrees that express in general terms how the references to the lists of identities will be implemented. To continue the previous example, the subtree headed by the node pointing to "range" might be

replaced by a subtree that would compile in a later stage to a loop that will execute 12 times.—

Please amend the paragraph at page 30, line 14, through page 30, line 18 to read as follows:

--In a preferred embodiment, the source, which is input as in block 7 of Fig. 5, is a natural language request. As discussed above for block 7, this request can be input into the computer in a variety of manners. The request, however, [is] may be in a natural language, such as English, French, or German, and [is] may be a request for information. As an example, one natural language request source could be "What is the population of the capital of Missouri?"--

IN THE CLAIMS:

Please cancel claims 1-20 add new claims 25-48 as follows:

25. (NEW) A method of processing user input comprising the steps of:

receiving the user input from a user;

matching concept representatives to the user input using experimentation to result

in a source tree; and

self-activating the source tree to interpret the user input.

26. (NEW) The method of claim 25, wherein the experimentation is performed recursively.

27. (NEW) The method of claim 25, wherein the concept representative includes

instructions for obtaining the source tree.

28. (NEW) A system for processing user input comprising:

means for receiving the user input from a user;

means for matching concept representatives to the user input using experimentation

to result in a source tree; and

means for self-activating the source tree to interpret the user input.

29. (NEW) The system of claim 28, wherein the experimentation is performed recursively.

30. (NEW) The system of claim 28, wherein the concept representative includes instructions for obtaining the source tree.

31. (NEW) A computer program product for processing user input comprising:
a computer usable medium having computer readable program code means
embodied in the computer usable medium for causing an application program to execute
on a computer system, the computer readable program code means comprising:
computer readable program code means for receiving the user input from a
user;

computer readable program code means for matching concept
representatives to the user input using experimentation to result in a source tree; and

computer readable program code means for self-activating the source tree to interpret the user input.

32. (NEW) The computer program product of claim 31, wherein the experimentation is performed recursively.

33. (NEW) The computer program product of claim 31, wherein the concept representative includes instructions for obtaining the source tree.

34. (NEW) A method of generating computer language code comprising the steps of:
receiving the user input from a user;
matching concept representatives to the user input using experimentation to result in a source tree; and
self-activating the source tree to convert the user input into computer language code.

35. (NEW) The method of claim 34, wherein the experimentation is performed recursively.

36. (NEW) The method of claim 34, wherein the concept representative includes instructions for obtaining the source tree.

37. (NEW) The method of claim 34, wherein the computer language code is in a high level language.

38. (NEW) The method of claim 34, wherein the computer language code is in a machine executable form.

39. (NEW) A system for generating computer language code comprising:
means for receiving the user input from a user;
means for matching concept representatives to the user input using experimentation
to result in a source tree; and
means for self-activating the source tree to interpret the user input.

40. (NEW) The system of claim 39, wherein the experimentation is performed recursively.

41. (NEW) The system of claim 39, wherein the concept representative includes instructions for obtaining the source tree.

42. (NEW) The system of claim 30, wherein the computer language code is in a high level language.

43. (NEW) The system of claim 39, wherein the computer language code is in a machine executable form.

44. (NEW) A computer program product for generating computer language code comprising:

a computer usable medium having computer readable program code means embodied in the computer usable medium for causing an application program to execute on a computer system, the computer readable program code means comprising:

computer readable program code means for receiving the user input from a user;

computer readable program code means for matching concept representatives to the user input using experimentation to result in a source tree; and

computer readable program code means for self-activating the source tree to interpret the user input.

45. (NEW) The computer program product of claim 44, wherein the experimentation is performed recursively.

46. (NEW) The computer program product of claim 44, wherein the concept representative includes instructions for obtaining the source tree.

47. (NEW) The computer program product of claim 44, wherein the computer

language code is in a high level language.

48. (NEW) The computer program product of claim 44, wherein the computer
language code is in a machine executable form.